PERFORMANCE IMPROVEMENTS IN VOICEMAIL TRANSCRIPTION

J. Huang, B. Kingsbury, L. Mangu, M. Padmanabhan, G. Saon, G. Zweig

IBM T. J. Watson Research Center P. O. Box 218,
Yorktown Heights, NY 10598
http://www.research.ibm.com/voicemail

ABSTRACT

In this paper we report recent improvements in word error performance on a voicemail transcription task. Last year, the speaker independent word error rate (WER) on the dev test set of the Voicemail Transcription task was reported at 35.45% [1]. This year, we report a relative 20% gain over this number. The improvements were obtained using several new algorithms and an increased amount of training data. In addition to benchmarking the performance of these algorithms on the Voicemail task, we have also evaluated them on the Switchboard task, and we report these results here as well. Finally, we also present the result of cross-domain experiments to evaluate the domain-independence of the constructed systems.

1. INTRODUCTION

In this paper we report recent improvements in transcribing conversational telephone speech, as typified by the Voicemail and Switchboard transcription tasks. These improvements are a result of some new algorithms and, in the case of Voicemail, also due to an increase in the amount of training data. In the following sections, we describe the contribution of several components to improving the word error rate. The Voicemail transcription task is described in [1] and represents samples of conversational telephone speech from a single speaker. The Switchboard task is described in several papers in [2] and represents samples of telephone conversations between two people.

One of the goals of speech recognition research is to design a domain-independent system (at least as far as the acoustic model is concerned) that can deal with various types of speech from the same category: for instance a system built on Switchboard should be able to provide the same performance on Voicemail as a system trained on Voicemail. Generally speaking, this has been an elusive goal, as the best performance is usually obtained by training the acoustic models on data drawn from the same domain as the test data. In this paper, we also evaluate the domain-independence of systems built with Voicemail and Switchboard training data.

We would like to acknowledge the support of DARPA under Grant MDA972-97-C-0012 for funding part of this work.

2. TRAINING/TEST DATA

<u>Voicemail</u>
The Voicemail training database now comprises 70 hours of speech, which corresponds to approximately 700k words of text. We will refer to this training database as T-VM1. The size of the testing vocabulary is 11k words. The development test set for this database comprises 43 messages (D-VM) and the evaluation test set (E-VM) comprises 62 messages.

<u>Switchboard</u>
We used 2378 of the 2438 Switchboard I conversations [2] as our training set, and the 19 conversations used in the 1997 Johns Hopkins Workshop as the test set. This represents around 200 hours of speech and 2 million words of text. We will refer to this training database as T-SWB1 and to the test database as E-SWB. The size of the vocabulary used for testing was 18k words.

3. SYSTEM DESCRIPTION

The speech recognition system uses a phonetic representation of the words in the vocabulary. Each phone is modelled with a 3-state left-to-right HMM. Further, we identify the variants of each state that are acoustically dissimilar by asking questions about the phonetic context in which the state occurs. The questions are arranged hierarchically in the form of a decision tree, and its leaves correspond to the basic acoustic units that we model. A feature vector is extracted every 10 ms, and we model the pdf of the feature vector for each leaf of the decision tree with a mixture of gaussians. The baseline feature vector is the Mel cepstrum augmented with its 1st and 2nd temporal derivatives. We will refer to this as the cepstral feature space. Some of the systems that we experimented with spliced together 9 frames of cepstra (the cepstra at the current frame; 4 frames before and after the current frame) and projecting the spliced feature vector

System	FSP	D	#L	#P	Trg
S-VM1 f433	Ceps	39	2313	134k	T-VM1
S-VM2 f708	Proj (1)	39	2313	134k	T-VM1
S-VM3 f810l	Ceps	39	2307	130k	T-VM1
S-VM4 f526	MSG	26	3527	154k	T-VM1
S-VM5 f844.v60	Ceps	39	2778	279k	T-VM1
S-VM6 f844.v70	Ceps	39	2778	279k	T-VM1
S-SWB1 f844.v28	Ceps	39	3140	275k	T-SWB1
S-SWB2 f901	Proj (2)	60	3140	275k	T-SWB1
S-VM7 f844.v50	Ceps	39	2778	279k	T-VM1 + T-SWB2
S-VM8 f844.v26	Proj (1)	39	2778	279k	T-VM1 + T-SWB1

Table 1: System description

down to a lower dimension. We will refer to this feature space as the projected feature space. Additionally, one system uses modulation filtered spectrogram (MSG) features [11].
We summarize the systems that we worked with in Table 1. The column FSP indicates the type of feature space, D indicates the dimensionality of the space, $\#L$ indicates the number of leaves, $\#P$ indicates the number of gaussians, Trg indicates the training data that was used to build the system.

4. FEATURE SPACE TRANSFORMATIONS

Linear discriminant analysis [3] is a standard technique for dimensionality reduction with minimal loss of discrimination information. However, the LDA formulation makes certain assumptions that are not always true. Chief among these is the assumption that all the classes have the same covariance matrix.

Let $\{x_i\}_{1\leq i\leq N}$ denote a sequence of D dimensional feature vectors, where each of the vectors belongs to a single class $j \in \{1,\cdots,J\}$. Let N_j, μ_j, Σ_j denote the sample count, mean and covariance of the j^{th} class. The class information may be condensed into two matrices called

$$\text{within-class scatter:} \quad W = \frac{1}{N}\sum_{j=1}^{J} N_j \Sigma_j$$

$$\text{between-class scatter:} \quad B = \frac{1}{N}\sum_{j=1}^{J} N_j \mu_j \mu_j^T - \overline{\mu}\overline{\mu}^T$$

The LDA objective function tries to find a PxD projection, θ, such that the ratio of the following determinants is maximized

$$J(\theta) = \frac{|\theta B \theta^T|}{|\theta W \theta^T|} \tag{1}$$

System	D-VM dev test	E-VM eval test	E-SWB
S-VM1	32.26	39.61	
S-VM2	30.23	35.26	
S-SWB1			45.69
S-SWB2			38.8

Table 2: HDA+MLLT

In [4] a HDA formulation was presented that modified the LDA objective function (1) to take into account the different covariance matrices of the different classes

$$\frac{|\theta B \theta^T|^N}{\prod_{j=1}^{J} |\theta \Sigma_j \theta^T|^{N_j}} \tag{2}$$

Taking the log of the above objective yields the HDA objective function

$$H(\theta) = \sum_{j=1}^{J} -N_j \log|\theta\Sigma_j\theta^T| + N\log|\theta B\theta^T| \tag{3}$$

The derivative of this objective may be derived to be

$$\frac{dH(\theta)}{d\theta} = \sum_{j-1}^{J} -2N_j(\theta\Sigma_j\theta^T)^{-1}\theta\Sigma_j + 2N(\theta B\theta^T)^{-1}\theta B \tag{4}$$

and quasi-Newton methods may be used to find the optimal solution.

The discrimination between classes provided in the HDA feature space requires the use of full-covariance gaussian models for the classes. This is generally too computationally expensive to be practical in most speech recognition systems; consequently, the models are replaced with gaussians that have diagonal covariances. If the HDA feature space is characterized by dimensions that are highly correlated, the modeling approximation inherent in the diagonal covariance assumption negates any benefit that the HDA may have. Therefore, we apply a further transformation (MLLT) that tries to diagonalize the HDA feature space [5]. The application of this transform does not change the HDA objective function value. The final feature space thus obtained will be referred to as the HDA+MLLT space. The classes that are used in the computation are the leaves of the decision tree.

The word error rate obtained on the D-VM, E-VM and E-SWB test sets for the cepstral and projected feature spaces are shown in Table 2. The HDA+MLLT space is seen to provide a relative improvement of 10-15% over the baseline cepstral space.

5. BOOSTING GAUSSIAN MIXTURES

Boosting is a technique for sequentially training and combining a collection of classifiers in such a way that the later classifiers make up for the deficiencies of the earlier ones. Many variants exist [7, 8] but all follow the same basic strategy. There is a sequence of iterations and at each iteration a new classifier is trained on a weighted set of the training examples. Initially, every example gets the same weight, but in subsequent iterations, the weights of hard-to-classify examples are increased relative to the easy ones. The outputs of the classifiers are then combined in such a way as to guarantee certain bounds on both training and testing error [8]. We report results here using an extension to AdaBoost that was presented in [6] and that allows for large speedups in training time. The extension was motivated by the scale of the problem, where we have tens of millions of labeled training pairs, thousands of classes, and and hundreds of thousands of gaussians that model the probability density of the classes.

The input to the AdaBoost algorithm is a set of labeled training pairs, (x_i, y_i), where x_i represents the features associated with the ith example and y_i is its label. In our application the x_i are acoustic feature-vectors and the y_i are context-dependent phone labels. At each iteration, t, a function $h_t(x, y)$ is learned that maps a feature/label pair into a number between 0 and 1. A weight, β_t, is assigned to each classifier, and the output of the composite classifier is given by

$$H(x,y) = \sum_t \left(\log \frac{1}{\beta_t} \right) h_t(x,y).$$

In our implementation the atomic classifiers are mixtures of gaussians with one mixture for each leaf.

In AdaBoost, each vector x_i is assigned a weight, $D_t(i, y)$, that is related to the probability with which x_i can be misrecognized as y. This implies that the complete classifier has to be designed in one step during the next iteration using gradient descent techniques. This process was simplified by the approximation in [6], which allowed the classifier to be designed in two steps. The weights over all classes for a given feature vector were summed up $D_t(i) = \sum_y D_t(i, y)$, and each feature vector now was associated with a single weight that is related to the probability of its having been misclassified during previous iterations. It is now possible to design gaussian mixtures independently for each class using only the weighted examples of the class. The i^{th} such mixture models the probability density function of x for class y_i, $p(x/y_i)$, and the classifier is now simply defined as $h_t(x,y) = \frac{p(x/y)}{\sum_k p(x/k)}$. The merit of this approach is that the process of classifier design can be parallelized and greatly speeded up. For details, the reader is referred to [6].

E-VM Test Set					
System	1st It.	2nd	3rd	4th	5th
S-VM1	39.61	39.48	39.15	39.10	38.92

Table 3: Boosting

D-VM Test Set		
System	Baseline	Consensus
S-VM2	30.23	28.86
S-VM3	33.7	31.24
S-VM4	42.4	41.6
Rover	29.2	28.5

Table 4: Consensus processing

The experimental results obtained by boosting the S-VM1 system are summarized in Table 3. The test set is the E-VM test set. The word error rates indicate a small but consistent improvement with increasing number of iterations.

6. CONSENSUS HYPOTHESIS PROCESSING

In all the experiments described earlier, the decoded hypothesis was taken to be the 1-best hypothesis in the search. Recently, [9] has shown that better performance can be obtained by considering all the hypotheses produced in the search and finding the "consensus hypothesis." In short, the word graph produced by the standard hypothesis search procedure is first converted into a chain-like structure by merging different paths in the graph. The components of the chain represent parallel sequences of words. The criterion for merging two paths in the graph is related to the time overlap between the paths and the phonetic similarity between the word sequences in the two paths. Subsequently, the most probable path (or word sequence) in each component of the chain is selected and the concatenation of these paths represents the consensus hypothesis. For further details, the reader is referred to [9].

We evaluated the performance of this technique on the E-VM test set with the S-VM2, S-VM3 and S-VM4 systems. Subsequently, we combined the consensus hypotheses of these three systems using ROVER [10]. The results are presented in Table 4. The baseline results refers to the 1-best hypothesis of the corresponding system

7. CROSS-DOMAIN EXPERIMENTS

In this section we examine the performance on the Switchboard test set using acoustic models trained on Voicemail

System	Training	Test	
Cross domain-Cepstral feature space			
		E-VM	E-SWB
S-VM5	T-VM1	39.5	62.2
S-SWB1	T-SWB1	53.5	45.8
Cross domain - Projected feature space			
S-VM6	T-VM1	36.3	57.3
S-SWB2	T-SWB1	46.75	38.5
Joint Training - Cepstral feature space			
S-VM7	T-VM1 + T-SWB2	41.7	48.7
Joint Training - Projected feature space			
S-VM8	T-VM1 T-SWB1	36.6	45.6

Table 5: WER performance for cross-domain condition

and vice versa. Note, however, that the language model and vocabulary were NOT mismatched. Superficially, as Voicemail and Switchboard both represent telephone bandlimited conversational speech, one would expect the performance on either test set to be independent of what database it is trained on, but the results show that this is not the case. The difference in performance also appears to depend on the feature space that is used. We present results here for several systems.

From Table 5, the performance degradation from the matched condition (shown underlined) due to a mismatch in the acoustic models ranges from 35-36% for the cepstral feature space to 29-49% for the projected feature space. The degradation appears to be worse for the Switchboard test set. Training the acoustic models on data from both domains does reduce the degradation to a large extent 6% for the cepstral feature space, to 1% for the projected feature space). The results show that the individual systems built on either training database are relatively domain-dependent, and that our current modeling techniques are not as robust as one might desire, and should be the focus of future algorithm development.

8. CONCLUSION

We report the following:
• overall reduction of 20% (relative) on Voicemail dev set
• results on the JHU 1997 Switchboard dev test set
• use of a novel linear projection (HDA+MLLT) that improves performance on the baseline cepstral feature space by 10-15% relative on both Voicemail and Switchboard
• use of boosting techniques for gaussian mixtures that yields 3% relative improvement
• use of a consensus hypothesis algorithm that provides a 3% relative improvement on both Voicemail and Switchboard
• cross-domain experiments that show the sensitivity of system performance to training data • the crude approach of making the system more robust by training on the union of all data sets does seem to work

9. ACKNOWLEDGEMENT

We would like to acknowledge CLSP at Johns Hopkins University and ISIP at Mississippi State University for their help in providing data sets to bring up our Switchboard system.

10. REFERENCES

[1] M. Padmanabhan, G. Saon, S. Basu, J. Huang, G. Zweig, ”Recent improvements on a Voicemail Transcription Task”, Proceedings of Eurospeech 1999.

[2] Proceedings of the Hub 5 Workshop, 1999.

[3] R. O. Duda and P. B. Hart, ”Pattern Classification and scene analysis”, Wiley, New York, 1973.

[4] G. Saon, M. Padmanabhan, R. Gopinath, S. Chen, ”Maximum likelihood discriminant feature spaces”, Proceedings of ICASSP 2000.

[5] R. Gopinath, ”Maximum likelihood modeling with gaussian distributions for classification”, Proceedings of ICASSP 1998.

[6] G. Zweig and M. Padmanabhan, ”Boosting gaussian mixtures in an LVCSR system”, Proceedings of ICASSP 2000.

[7] Y. Freund and R. Schapire, ”Experiments with a new boosting algorithm”, Proceedings of Intl Conference on Machine Learning, July 1996, Morgan Kaufman.

[8] R. Schapire, Y. Freund, P. Bartlett, W. S. Lee, ”Boosting the margin: A new explanation for the effectiveness of voting methods”, Annals of Statistics, 26(5): 1651-1686, 1998.

[9] L. Mangu, E. Brill and A. Stolcke, ”Finding consensus among words: lattice-based word error minimization”, Proceedings of Eurospeech 1999.

[10] J. G. Fiscus, ”A Post-Processing System to Yield Reduced Word Error Rates: Recogniser Output Voting Error Reduction (ROVER)”, Proceedings of IEEE ASRU Workshop, pp. 347-352, Santa Barbara, 1997.

[11] B. E. D. Kingsbury, N. Morgan and S. Greenberg, ”Robust speech recognition using the modulation spectrogram”, Speech Communication, 25(1-3), pp 117-132, 1998.